1 Lab 02

Question 1. The program enters an infinite loop and eventually terminates because the recursion limit has been exceeded. This is because is is never checked, whether the agent has seen the current state already. Adding the following line at the indicated position in the code solves the problem:

```
if perceived_state_representation in encountered_states:
    return False, None
```

Question 2. Let S and A denote the state and action sets, respectively. We denote the initial state by $s_I \in S$ and the set of goal states by $S_G \subseteq S$. Then, Towers of Hanoi with $N \in \mathbb{N}$ disks can be formalized as follows.

- A state can be represented via the content of its three rods. The order is fixed by rule 3 and thus need not be encoded. In summary, $S = \{(s_1, s_2, s_3) \mid s_i \subseteq \{1, \ldots, N\} \text{ for } i \in \{1, 2, 3\} \text{ and } \bigsqcup_{i=1}^{3} s_i = \{1, \ldots, N\}\}$.

  (An alternative representation is $\tilde{S} = \{1, 2, 3\}^N$, which can be interpreted as a mapping from each disk to the corresponding rod it sits on.)

- The initial state is $s_I = (\{1, \ldots, N\}, \varnothing, \varnothing)$.
- The goal states are $S_G = \{(\varnothing, \{1, \ldots, N\}, \varnothing), (\varnothing, \varnothing, \{1, \ldots, N\})\}$.
- Let $a_{i,j}$ denote the action of removing the top disk of rod i and placing it at the top of rod j. Then, the set of actions is $A = \{a_{i,j} \mid i, j \in \{1, 2, 3\} \text{ and } i \neq j\}$.
- If the ith rod is empty, all actions of the form $a_{i,j}$ are unavailable. If the top disk of rod j is smaller than the one on rod i, the action is also not available. Hence,
  $\textsc{Actions}(s) = \{a_{i,j} \mid i, j \in \{1, 2, 3\}, s_i \neq \varnothing \text{ and } \min s_j > \min s_i\}$.
- If action $a_{i,j}$ is available in state s, we have $\textsc{Result}(s, a_{i,j}) = (s'_1, s'_2, s'_3)$, where $s'_i = s_i \setminus \{\min s_i\}$ and $s'_j = s_j \cup \{\min s_i\}$. The remaining rod, call it k, is unchanged, so $s'_k = s_k$.

Note: This is one possible formalization and there may be others.

Question 3. The depth of solutions found by BFS is always optimal. The pattern that emerges is a solution depth of $2^N - 1$.

Remark: Indeed, this can be proven by induction. Given $N = 1$ disk, the optimal solution clearly has depth $1 = 2^1 - 1$. Assume that we have shown this fact up to a certain N. To show that it holds for $N + 1$, first move the top N disks to the second or third rod – which can be done optimally in $2^N - 1$ steps. Then, move the disk $N + 1$ onto the empty rod. Finally, move the stack of N disks back onto the one containing disk $N + 1$ – which once again can be done optimally in $2^N - 1$ steps. In total, we have performed $2 \cdot (2^N - 1) + 1 = 2^{N+1} - 1$ steps.

Question 4. DLS with a depth-limit of 1000 takes 364 steps. This is much larger than the optimal depth of 63. This is, however, no contradiction to the previous answer, because DFS is not guaranteed to find an optimal path.

Question 6. The length of the shortest path is 5. The path goes as follows:

Out of memory → Personal computer → Typewriter → General Motors → List of rooftop photovoltaic installations → Solar power in Germany.

This solution has been obtained with BFS. Possible implementation:

```
from collections import deque

def length_of_shortest_path(start_page: str, end_page: str, wikigraph: Wikigraph):
    start_id = wikigraph.get_id(start_page)
    end_id = wikigraph.get_id(end_page)

    Q = deque([start_id])
    parent = {start_id: None}
    seen = set([start_id])
    while Q:
        v = Q.pop()

        if v == end_id:
            path = []
            while v:
                path.append(wikigraph.get_name(v))
                v = parent[v]
            print(' -> '.join(path[::-1]))
            return len(path)

        for w in wikigraph.get_links(v):
            if w in seen:
                continue
            Q.appendleft(w)
            seen.add(w)
            parent[w] = v
    return -1

wikigraph = Wikigraph()
length_of_shortest_path('Out of memory', 'Solar power in Germany', wikigraph)
```